Markov_Chains

Eduardo de Jesús Cuellar Chávez

3/3/2022

¿Por qué son importantes las cadenas de Markov?

Muchos procesos estocásticos son usados para modelar el comportamiento de activos financieros *(Pequeño spoiler del final del curso)* y otros modelos usados en análisis de supervivencia (Modelos multiestado de riesgos competitivos), seguros (Vida múltiple), marketing (Visitas de una página o perfil de internet), lo cual hace relativamente fácil simular estos procesos.

En este pequeño `Rmarkdown` presentaremos una breve introducción a la simulación de estas famosas **Cadenas de Markov**, ¡Sí, como el perrito del cartel del curso!

Figure 1: ¡Markov!

Haremos énfasis en las cadenas con estados y tiempos discretos, que son los que hemos estado viendo en clase.

¿Por qué es de utilidad la simulación?

Ayuda a aterrizar los conceptos teóricos en algo práctico, además de que nos ayuda bastante a visualizar qué es lo que está pasando "en la práctica", con la teoría.

Pequeño recordatorio

Sea $\mathbb{S}$ un conjunto discreto el espacio de estados, y π la distribución inicial de la cadena, así como P_{ij} la probabilidad de ir del estado i al j, con $i, j \in \mathbb{S}$.

Decimos que un proceso estocástico $\{X_n : n \in \mathbb{N}\}$ es una cadena de Markov si cumple que $\forall n \geq 0$ y estados $i, j \in \mathbb{S}$:

$$\mathbb{P}(X_{n+1} = j | X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = i_{n-1}, X_n = i) = \mathbb{P}(X_{n+1} = j | X_n = i) = P_{ij}$$

Recordemos que:

P_{ij} denota la probabilidad de que la cadena, **estando actualmente en tiempo n en el estado i**, se mueva en el siguiente paso ("unidad de tiempo"), al estado j. A esta probabilidad también se le conoce como **probabilidad de transición de un paso**

Si unimos todas estas probabilidades para cada i en un vector, y los acomodamos como renglones de una matriz, **obtenemos una matriz cuadrada** $\mathbb{M} = P_{ij} \forall i, j \in \mathbb{S}$. A esta matriz la llamamos **matriz de transición de un paso**, ya que cuando la cadena deja el estado i, debe moverse a algún estado $j \in \mathbb{S}$

Con lo comentado anteriormente, es fácil darnos cuenta de que hay una distribución para cada estado inicial a tiempo n, es decir, cada renglón debe sumar uno:

$$\sum_{j \in \mathbb{S}} P_{ij} = 1$$

Con esto, estamos asumiendo que las probabilidades **no** dependen del tiempo, sino **únicamente del estado en el que nos encontramos actualmente**

Algoritmo para simular

1. Definir la distribución inicial π, así como la matriz de transición $\mathbb{M}$, así como definir el espacio de estados.
2. Obtener una muestra de tamaño 1 de la distribución inicial y asignársela a la variable X_0
3. Hacemos un subsetting para quedarnos con el renglón donde estamos parados (es decir, del renglín i-ésimo)
4. Usaremos la función sample para obtener los estados
5. Repetimos desde 3 hasta que hayamos realizado las n iteraciones que queremos

Implementación

```
# pi es la distribución inicial
# m la matriz de transición a un paso
# n el número de simulaciones
# s el vector de estados
Markov<-function(pi,m,s,n){
  #Número posible de estados
  num_estados <- nrow(m) #o la longitud del vector o bien: length(s)
  #Estados que va tomando
  estados <- numeric(n)
  #inicializamos la primera entrada
  estados[1] <-sample(s, 1, prob = pi)
  #Ahora usamos la matriz de transición
  for(t in 2:n) {
    # Obtenemos el vector de probabilidades correspondiente
```

```
    #a dicho estado en el que caímos
    p  <- m[estados[t-1], ]
    # una muestra al azar de los estados
    estados[t] <-  sample(s, 1, prob = p)
  }
  return(estados)
}
```

Ejemplo:

```
s<-c(1,2,3,4)
pi<-c(1/4,1/4,1/4,1/4)
p1<-c(0,0,1/3,2/3)
p2<-c(0,1/2,0,1/2)
p3<-c(1/3,1/3,1,1/3)
p4<-c(1/2,0,1/2,0)
n=116
m<-matrix(c(p1,p2,p3,p4),nrow=4,ncol=4,byrow = T);m
```

```
##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.0000000 0.0000000 0.3333333 0.6666667
## [2,] 0.0000000 0.5000000 0.0000000 0.5000000
## [3,] 0.3333333 0.3333333 1.0000000 0.3333333
## [4,] 0.5000000 0.0000000 0.5000000 0.0000000
```

```
simulaciones<-Markov(pi,m,s,n)
```

Visualizamos:

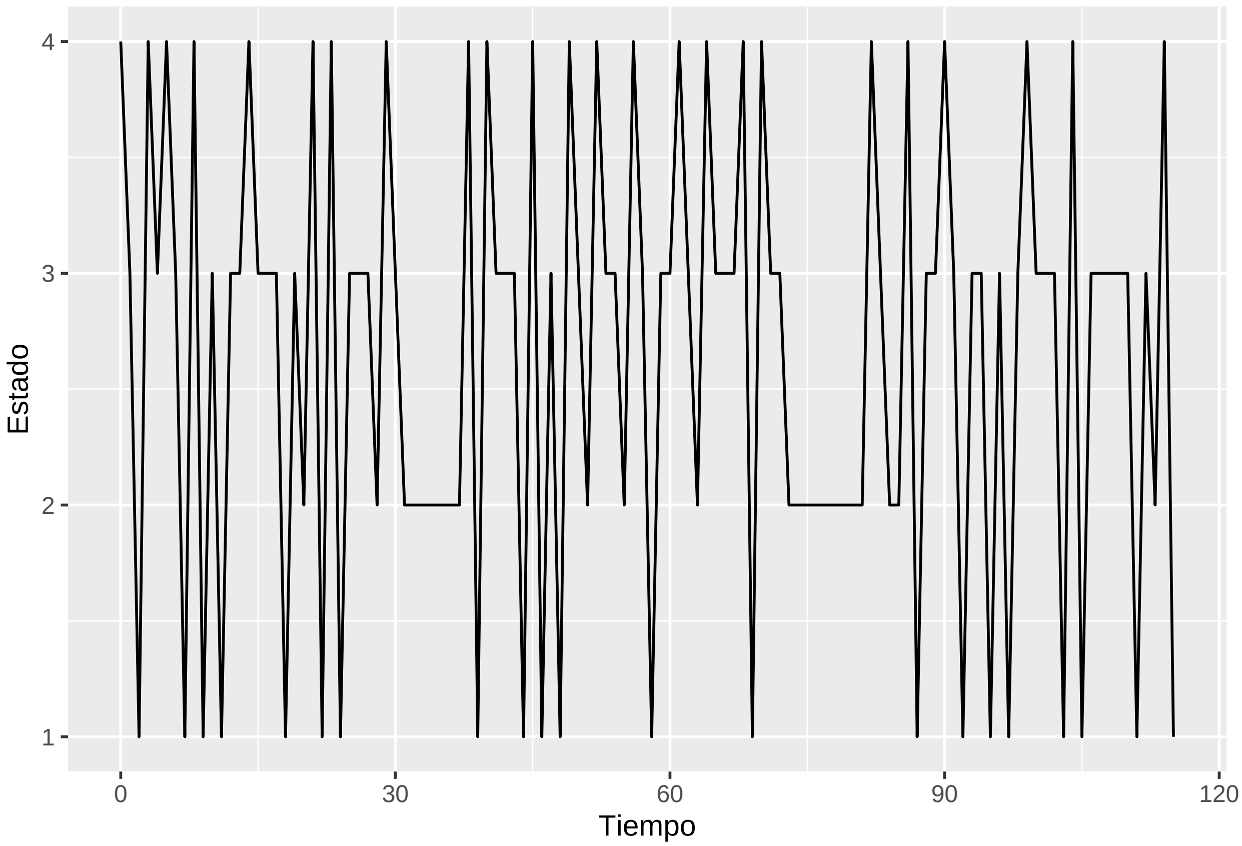
```
library(ggplot2)
tiempo<-seq(from=0,to=(n-1),by=1)
datos<-data.frame("Tiempo"=tiempo,"Estado"=simulaciones);datos
```

```
##     Tiempo Estado
## 1        0      4
## 2        1      3
## 3        2      1
## 4        3      4
## 5        4      3
## 6        5      4
## 7        6      3
## 8        7      1
## 9        8      4
## 10       9      1
## 11      10      3
## 12      11      1
## 13      12      3
## 14      13      3
## 15      14      4
## 16      15      3
## 17      16      3
## 18      17      3
```

```
## 19         18          1
## 20         19          3
## 21         20          2
## 22         21          4
## 23         22          1
## 24         23          4
## 25         24          1
## 26         25          3
## 27         26          3
## 28         27          3
## 29         28          2
## 30         29          4
## 31         30          3
## 32         31          2
## 33         32          2
## 34         33          2
## 35         34          2
## 36         35          2
## 37         36          2
## 38         37          2
## 39         38          4
## 40         39          1
## 41         40          4
## 42         41          3
## 43         42          3
## 44         43          3
## 45         44          1
## 46         45          4
## 47         46          1
## 48         47          3
## 49         48          1
## 50         49          4
## 51         50          3
## 52         51          2
## 53         52          4
## 54         53          3
## 55         54          3
## 56         55          2
## 57         56          4
## 58         57          3
## 59         58          1
## 60         59          3
## 61         60          3
## 62         61          4
## 63         62          3
## 64         63          2
## 65         64          4
## 66         65          3
## 67         66          3
## 68         67          3
## 69         68          4
## 70         69          1
## 71         70          4
## 72         71          3
```

```
## 73       72       3
## 74       73       2
## 75       74       2
## 76       75       2
## 77       76       2
## 78       77       2
## 79       78       2
## 80       79       2
## 81       80       2
## 82       81       2
## 83       82       4
## 84       83       3
## 85       84       2
## 86       85       2
## 87       86       4
## 88       87       1
## 89       88       3
## 90       89       3
## 91       90       4
## 92       91       3
## 93       92       1
## 94       93       3
## 95       94       3
## 96       95       1
## 97       96       3
## 98       97       1
## 99       98       3
## 100      99       4
## 101     100       3
## 102     101       3
## 103     102       3
## 104     103       1
## 105     104       4
## 106     105       1
## 107     106       3
## 108     107       3
## 109     108       3
## 110     109       3
## 111     110       3
## 112     111       1
## 113     112       3
## 114     113       2
## 115     114       4
## 116     115       1
```

```
#Podemos ver cómo va saltando
ggplot(datos,aes(x=Tiempo,y=Estado))+geom_line()
```



4
3
2
1
Estado
0
30
60
90
120
Tiempo